Toxic Comment Classification Using DistilBert

By: Hesham Samir Abd Elaziz Hagag 2000513

Contents

1 Introduction:

Nowadays users leave numerous comments on different social networks, news portals, and forums. Some of the comments are toxic or abusive. Due to the number of comments, it is unfeasible to manually moderate them, so most of the systems use kind of automatic discovery of toxicity using machine learning models.[2]

Researchers used different methods to solve this problem as shown in the following table.

Machine learning method	Number of papers
Convolutional neural network (CNN)	12
Logistic regression classifier	9
Bidirectional long short-term memory (BiLSTM)	8
Bidirectional Gated Recurrent Unit Networks (Bidirectional GRU)	6
Long Short Term Memory (LSTM)	5
Support Vector Machine (SVM)	5
Bidirectional Encoder Representations from Transformers (BERT)	4
Naive Bayes	4
Capsule Network	3
Random Forest	2
Decision tree	2
KNN classification	2
Gated Recurrent Unit (GRU)	2
Extreme Gradient Boosting (XGBoost)	2
Recurrent Neural Network (RNN)	2
Bi-GRU-LSTM	1
Gaussian Naive Bayes	1
Genetic Algorithms (GA)	1
Partial Classifier Chains (PartCC)	1

Figure 1 Machine learning methods used in toxic comments classification

Different transformers have recently shown a superior performance in many natural language processing tasks. In this document I will use Hugging Face DistilBert Transformer to classify toxic comments.[1]

2 BERT Model

Stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models, BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).[3]

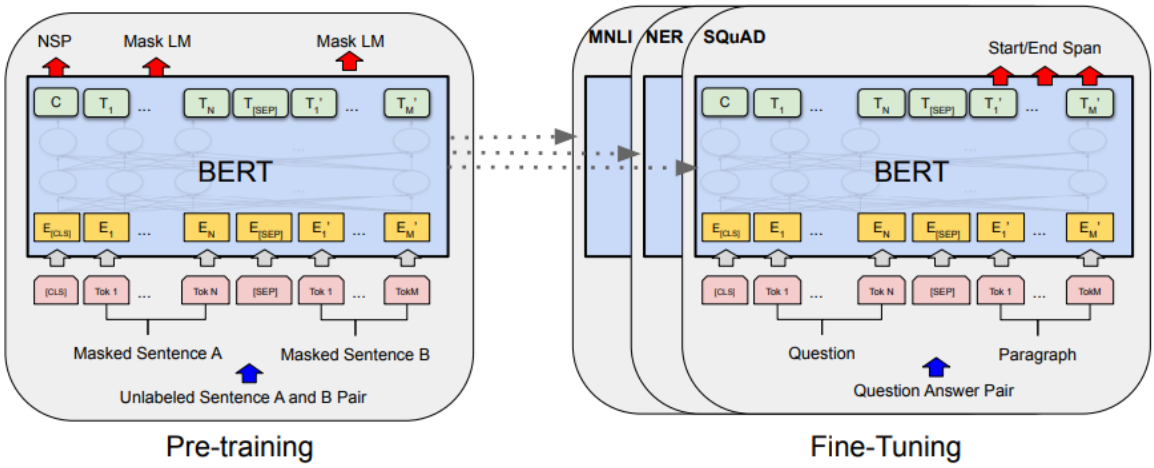

Figure 2 Overall pre-training and fine-tuning procedures for BERT.

3 DistilBERT Model

DistilBERT is a small, fast, cheap and light Transformer model trained by distilling BERT base. It has 40% less parameters than bert-base-uncased, runs 60% faster while preserving over 95% of BERT's performances as measured on the GLUE language understanding benchmark.[4]

4 Happy Transformers

Happy Transformer is built on top of Hugging Face's transformers library and allows programmers to implement and train Transformer models with just a few lines of code.

5 Dataset

The used data set consists of a large number of Wikipedia comments which have been labeled by human raters for toxic behavior. The types of toxicity are:

- toxic
- severe_toxic
- obscene
- threat
- insult
- identity_hate

I used only the first label(toxic). This dataset is split into 30,000 sample for training and 30,000 sample for testing as shown in the following figures.

text	label
Pan American Union]] (Final Act of the Second Meeting of the Ministers of Foreign Affairs of the American Republics at Habana, Cuba, July 30, 1940)	0
"	0
Mama's Family	0
"	0
"	0
"	0
! Note that using subst instead of transclusion makes it so that the IP addresses have to be manually added to the Category:AOL IP addresses - rather than just using what links here on the tem	0
"	0
Nobody said that. But you have a COI and you are removing content clearly referenced.	0
"::::::::BTW I didn't imply all sociologists were necessarily scientifically illiterate so please do not slander me.	0
RE: Been a long time...	0
"	0
I would like to say that the category at the bottom does not apply to Eco. He was a great Wikipedian, but, alas, his bad side got the best of him. If you want to know more, read . Thank you. (tal	0
":::: Steve, sorry I had to revert your last edit. The wording ""that Jews come to believe that Jesus..."" looks very POV to me - as if this is the truth that they indeed should eventually ""come	0
Ludwig von Mises Institute	0
Neutrality of the article & corrections	0
HEAAAGGGGGGGHHHHHH!!!!!!!!	1

Figure 3 Training set sample.

text	label
Explanation	0
D'aww! He matches this background colour I'm seemingly stuck with. Thanks. (talk) 21:51, January 11, 2016 (UTC)	0
Hey man, I'm really not trying to edit war. It's just that this guy is constantly removing relevant information and talking to me through edits instead of my talk page. He seems to care m	0
"	0
You, sir, are my hero. Any chance you remember what page that's on?	0
"	0
****	1
Your vandalism to the Matt Shirvington article has been reverted. Please don't do it again, or you will be banned.	0
Sorry if the word 'nonsense' was offensive to you. Anyway, I'm not intending to write anything in the article(wow they would jump on me for vandalism), I'm merely requesting that it	0
alignment on this subject and which are contrary to those of DuLithgow	0
"	0
bbq	0

Figure 4 Test set sample.

6 Code

Data set and code can be found in this repo. Here is a snapshoot from the code that I used to train hugging face DistilBERT model.

```
#Check GPU

import tensorflow as tf

device_name = tf.test.gpu_device_name()

if device_name != '/device:GPU:0':

  raise SystemError('GPU device not found')

print('Found GPU at: {}'.format(device_name))


#Install and import happy transformers

pip install happytransformer

from happytransformer import HappyTextClassification


#Create Model

tarnsformer_model = HappyTextClassification(model_type="DISTILBERT", model_name="distilbert-base-uncased", num_labels=2)


#Train Model

from happytransformer import TCTrainArgs

args = TCTrainArgs(num_train_epochs=1, learning_rate = 0.1, batch_size = 1)

with tf.device('/device:GPU:0'):

  tarnsformer_model.train("../input/train3/train-toxic.csv", args=args)


#Evaluate

tarnsformer_model.save("./dBert")

with tf.device('/device:GPU:0'):

    result = tarnsformer_model.eval("/kaggle/input/testdata3/test-toxic.csv")

print(result)
```

7 Adjusting Model Parameters

7.1 Learning rate

Learning rate is how much the model's weights are adjusted per step. Too low and the model will take a long time to learn or get stuck in a suboptimal solution. Too high can cause can divergent behaviors.

I tried different values for the learning rate which are 0.001, 0.01, 0.1, 0.5, 0.9. Then I calculated loss on the test set and time of training. I also made a plot for the loss during training steps. I summarized results in the following sections.

7.1.1 Learning rate = 0.001

Loss of test dataset = 0.531887412071228

Time = 26:56

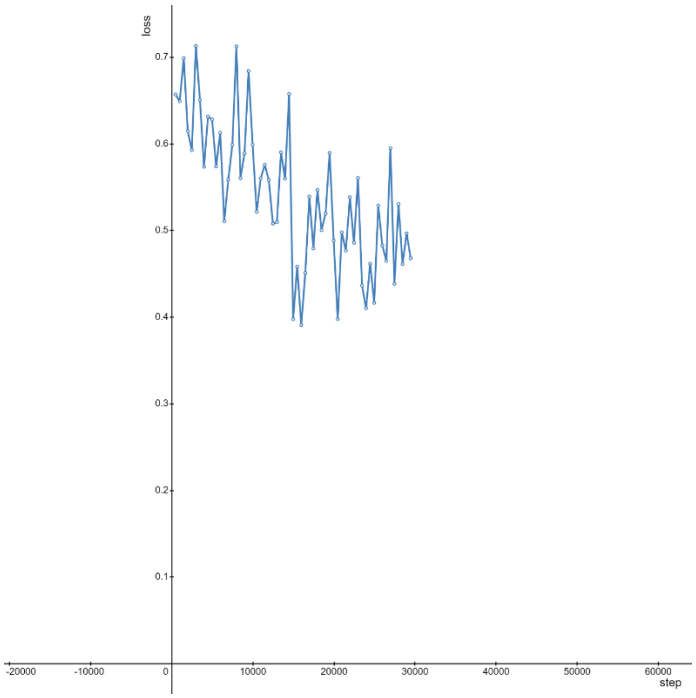

Figure 5 Graph of training loss vs training step number when learning rate is 0.001.

7.1.2 Learning rate = 0.01

Loss of test dataset = 0.4855969250202179

Training time = 26:44

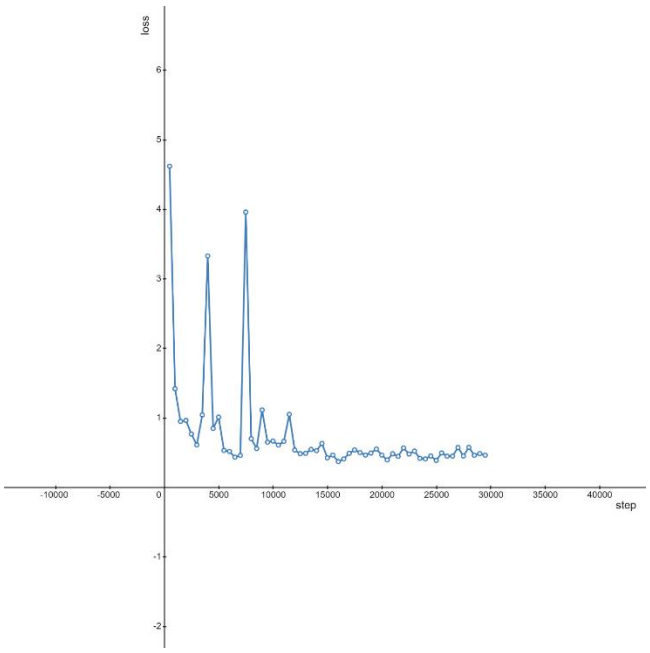

Figure 6 Graph of training loss vs training step number when learning rate is 0.01.

7.1.3 Learning rate = 0. 1

Loss of test dataset = 0.32595914602279663

Training time = 26:41

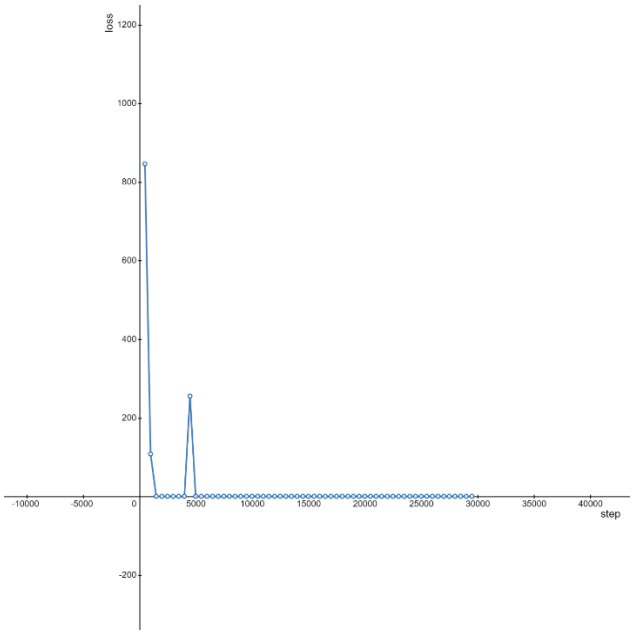

Figure 7 Graph of training loss vs training step number when learning rate is 0.1.

7.1.4 Learning rate = 0. 5

Loss of test dataset = 0.3243422210216522

Training time = 26:52

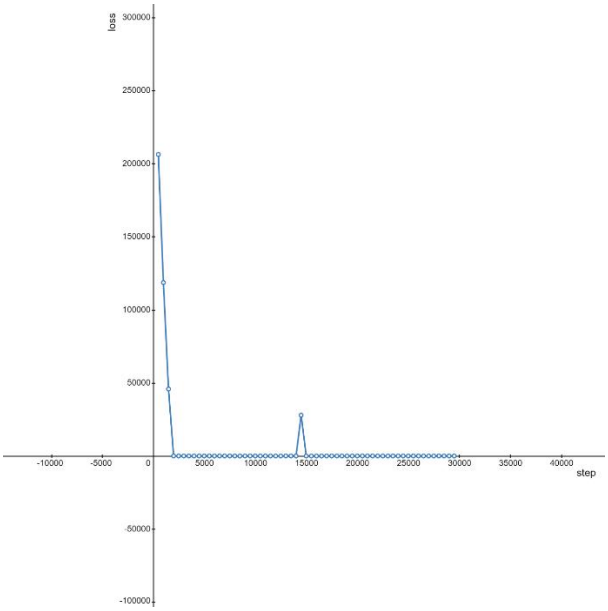

Figure 8 Graph of training loss vs training step number when learning rate is 0.5.

7.1.5 Learning rate = 0.9

Loss of test dataset = 0.3251972198486328

Training time = 26:44

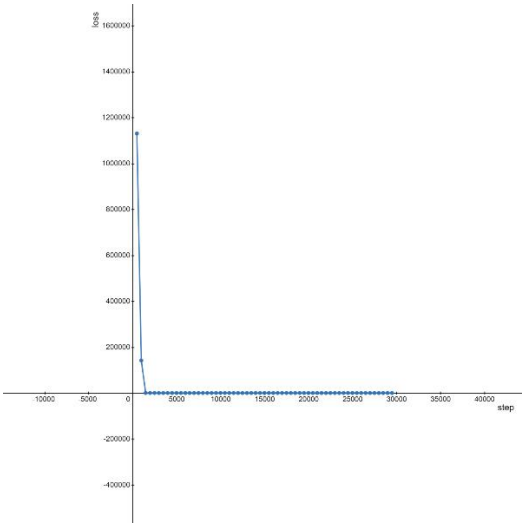

Figure 9 Graph of training loss vs training step number when learning rate is 0.9.

As we can see, the optimum learning rate value is 0.5. In the following section I will use this learning rate with different values of other parameters

7.2 Batch size

Batch size is the Number of training examples used per iteration.

7.2.1 Batch size = 5

Loss of test dataset = 0.32186710834503174

Training time = 18:41

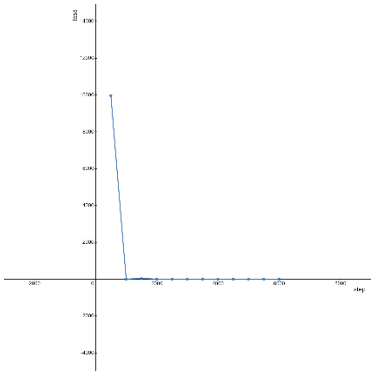

Figure 10 Graph of training loss vs training step number when batch size is 5.

Conclusion:

Since we have 30 000 training sample and we are taking 5 samples per each step, so we have 30 000/5 step which are 6000.

Increasing batch size decreases the loss of the test set and consumes around 1.5 less time.

7.2.2 Batch size = 10

Loss of test dataset = 0.32192695140838623

Training time = 15:06

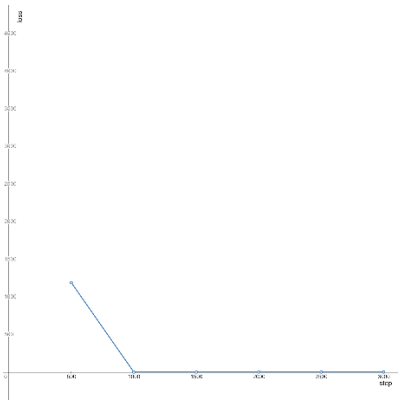

Figure 11 Graph of training loss vs training step number when batch size is 10.

Conclusion:

When batch size is 10, Training time decreases but loss increased slightly. So, for the coming trials I will use batch size of 5.

7.3 Number of training epochs

The number of times the training data is iterated over.

7.3.1 Training epochs = 3

Loss of test dataset = 0.3218967318534851

Training time = 52:35

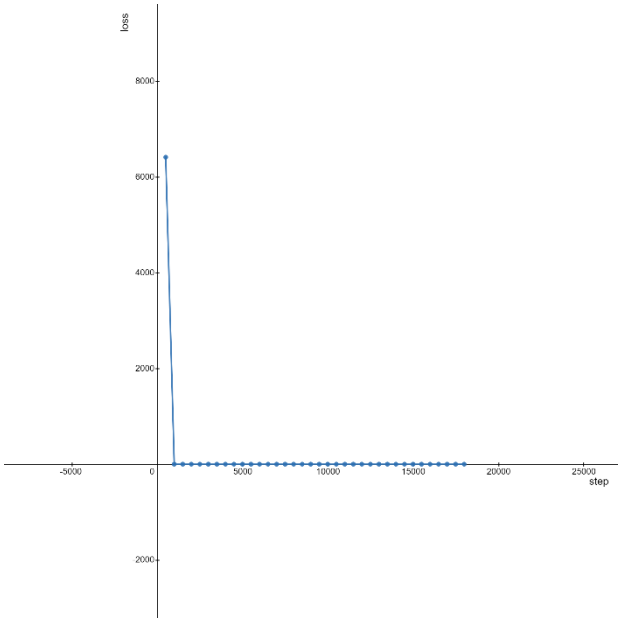

Figure 12 Graph of training loss vs training step number when epochs number is 3.

Conclusion:

Increasing number of epochs to 3 increases the loss slightly and it consumes more time than using 1 epoch.

8 Conclusion

DistilBert Shows good results in toxic comment classification. The optimal results can be produces using the following parameters:

- Learning rate = 0.5
- Batch size = 5
- Epochs number = 1

This model produces loss equals to 0.32186710834503174 and trained in 18 minutes and 41 seconds.

9 Table of Figures

10 References:

[1] Machine learning methods for toxic comment classification: a systematic review by Darko Androcec

[2] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/overview

[3] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova.

[4] Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT by Victor Sanh